LISTing Newsletter

Newseltter of the Long Island Sinclair/Timex Userds Group
(Incorporating N.Y.T.S.E.)

Issue January 1992

Welcome to 1992!

Post Script on Timex?

Savings and Load

LIST
5 Peri Lane
Valley Stream, NY 11581

TO:

DON LAMBERT JAN/93
1301 KIBINGER PL
AUBURN IN 46706-3010

FIRST CLASS MAIL
Dated Meeting Notice
February 9 1992

LIST OFFICERS
****
President.....Harvey Rait
Treasurer.....Robert Malloy
Cor. Secretary...John Pazmino
Librarian.....Tom Skapinski
LISTing Editor...Alvin Brandon
****

ease send inquiries to:
LIST
Mr. Harvey Rait
5 Peri Lane
Valley Stream, NY 11581

Please send submissions to:
Alvin Brandon
367 South 5th Street Apt 1E
Brooklyn, NY 11211
****
N.Y.T.S.E.
****
NYTSE meets on Monday the week after the LIST meeting at:
Miss Kim's Restaurant
Park Avenue South
(Bet. 21st and 22nd Streets)
Meetings start at 7:30 PM
****
Coming Events:
January 12, 1992 - LIST meeting at 2:00 PM

January 20, 1992 - NYTSE meeting at 7:30 PM
****

Meeting Minutes
December 8, 1991
****
Harvey called the meeting to order at 2:20 PM. Attendance was only 8 due to calendar glitch and the holidays.

en Lang demoed a QL program that rcles through a set of video frames to ake animation. The program, called vie Demo, can capture frames from a eo interface attached to the QL bus.

Skapinski noted that membership stable at about 80, and actually grows y 2 or 3 each year.

Bob Gilder noted that the QL Gold Card now sells for US$350 and incorporates the facility for 3.2mb disk drives. The QL Rebel I/F now comes with either a 10mb Hard Drive or a 600mb CD Rom Drive. for US$1000.

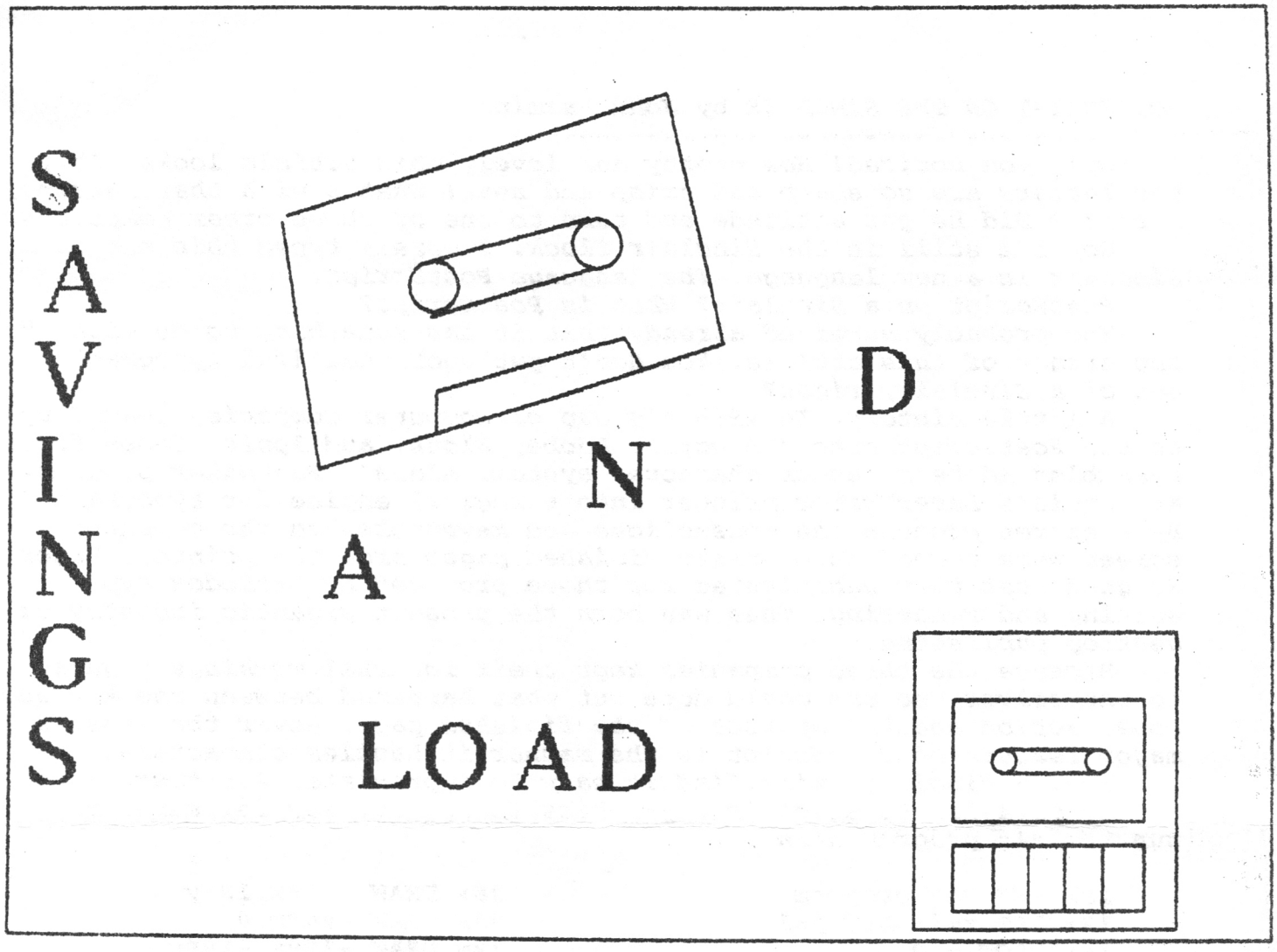
John Pazmino and Ken Lang reminded that Your Sinclair and QL World welcome U.S. subscribers. Your Sinclair, John pointed out, comes each month with a cassette of applications, about six or seven on each cassette. However, due to shipping by air and the pound/dollar ratio, subs are pricey, about $70.00 per year.

Fred Stern and Steve Kaye reported that T/S 1000s are still in production by special order, mainly as uncased units. They are being bought up by the thousands for industrial and laboratory work requireing digital control and monitoring.

Ken Lang reports that QDOS is still owned by Amstrad who have no intent to sell or lisence it, being that the QL is out of production. Thor still makes a QLike machines under pre-Amstrad agrements, but are losing share rapidly to IBM in businesses. There is a QDOS clone that is sold only to existing QLs and not to install into all new machines. From all this, the total number of QLs in the world is frozen and only used or remainder equipment is available.

Harvey Rait remarked that from his work with new immigrants he sees that Russian (nee Soviet) arrivals already are well versed in Sinclair. They tell him that Sinclair is the walkway 8-bit computer in the CIS and that it is installed in many Russian schools.

The January LIST ... semi...



At last a safe place to invest your money!

Get your copy today and end all your cassette problems once and for all!!!!
Contact LIST for more information.

LIST
c/o Harvey Rait
5 Peri Lane
Valley Stream, NY 11581

POSTSCRIPT ON THE SINCLAIR by John Pazmino

Oy!, you noticed! How pretty and lovely this article looks! Oh!, the letters are so sharp and crisp and neat! What's with that Pazmino, already? Did he get attitude and turn to one of those other computers?

No, I'm still in the Sinclair flock. I merely typed this out on my Sinclair in a new language. The language PostScript.

PostScript on a Sinclair? What is PostScript?

You probably surmised already that it has something to do with the appearance of this article. You don't get such beautiful typography out of a Sinclair, right?

A little history. In 1985 a group of computer companies teamed up to sic PostScript onto the world: Adobe, Aldus, and Apple. These firms assembled Adobe's vector character system, Aldus's PageMaker program, and Apple's LaserWriter printer into a magical engine for typography. By a secret process the mouseclicks and keystrokes on the computer screen were turned into snazzy finished pages from the printer. Pages so good that they substituted for those produced by orthodox type setting and rendering. Thus was born the present gigantic industry of desktop publishing.

Because the three companies kept their internal workings tightly to themselves, no one could dope out what happened between the key and mouse action and the spitout of the finished page. Never the less, a major feature of PostScript is the manner it handles characters.

You, a Sinclair, will find it easy to appreciate. For there is a strong parallel to Sinclair's graphics functions. Key in, save, and run the "A" program here.

```
10> REM "A" program
12> LET x=1: LET y=1
14> PLOT 0,0
16> DRAW 10*x,5*y
18> DRAW 40*x,60*y
20> DRAW 40*x,-60*y
22> DRAW 10*x,-5*y
24> DRAW -25*x,0
26> DRAW 10*x,5*y
28> DRAW -12*x,18*y
30> DRAW -46*x,0
32> DRAW -12*x,-18*y
34> DRAW 10*x,-5*y
36> DRAW -25*x,0
38> PLOT 29*x,27*y
40> DRAW 42*x,0
42> DRAW -21*x,32*y
44> DRAW -21*x,-32*y
```

You get a [sort of] neat capital letter "A" on the screen. It is made of PLOT and DRAW operations. Fundamentally there is no difference between this "A" and an ordinary picture made from the same PLOT/DRAW functions. From your immersion in the Latin environment you say this is a letter "A" and not a tree or a goat.

With the "A" still printed on the screen do a screensave of it. You just captured a block of the memory containing the image of the "A". Each pixel on the screen is mapped one-to-one to a bit in the memory. You in truth created a bitmapped file of the "A". The original "A" program is a vector or outline file of the same letter "A".

Do NEW. Now you want to restore the "A" to the screen. You can load and run the "A" program or you can load the "A" screen. The "A" screen -- the bitmapped file -- is way the longer of the two at several kilobytes. The vector program is only a few hundred bytes long. Hence, one distinction between the bitmapped file and the outline file is the greatly larger size of the former.

There's more. Edit line 12 so x=0.6 and y=1.2, then run the

program. See?, you get a slimmer and taller "A". Try other values for x and y and get all sorts of stretched and sized "A"s. You may hit against an 'out of range' error if the letter spills off of the screen, but apart from that you can make any "A" you like.

Not so in the bitmapped scheme. You got to have a new and other file for each variety of "A" you want. If your text calls for eight sizes of "A" you need eight bitmapped files of that "A". Yet each variety of "A" is generated by a simple edit of the one and same "A" outline program. Ergo, in the place of a battery of bitmap files you need but a one vector file.

There's a further and monumental difference between the two systems of character. The bitmapped character must be designed to coordinate with the peculiar printer you got. An "A" made to look good on a 9-pin Epson may look lousy on a 9-pin Citizen or even a 24-pin Panasonic. This is due to the diverse grid, shape, and spacing of the pixels among the printers.

The vector method employs a phantom mathematical coordinate system, unattached to any particular output device; it is 'device independent'. The drawing instructions are sent to the printer, which does whatever is necessary to reproduce the character the way it's supposed to look. You glimpsed this already in the "A" program. You plugged in x and y values in full disregard of the pixelgrid. The coordinate of any point in the "A" can well be a noninteger number of pixels. Yet the "A" looks rather nice anyway. (By the way, did you know that Sinclair does round off nonintegral coordinates?)

PostScript uses the outline or vector scheme of character generation. It is device-independent. That is, once you work out the drawing instructions for the "A" you can apply them to any printer and get a fair and proper realization of the "A".

In the outline system you can -- no kidding -- code the entire Algonquin pubescence oath on a pinhead. Or an "SOS' from Earth large enough to read from Jupiter. Try it with the "A" program. Let x=1e-10, y=1e-10. See? A [very] miniature "A". Uh, wait a minute; that's just a dot there. Go and look closer. Yep, it's a dot. Yet Sinclair really did carry out mathematically the scaling and that dot just happens to be a set of individual coordinates all rounded to (0,0). If you send that program to a microphotoetcher, used for inscribing computer chips, and look at the output thru a microscope, lo!, there before your eyes is a perfectly formed "A". And this is a good thing.

Such microscopic size of character is impossible in the bitmap system. There is no way to form a recognizable letter out of a single pixel! At the other extreme there is no [at least for a year or so] computer big enough to hold the squillions of bits for the "SOS".

Now the "A" program has one style of "A". You can redefine the basic shape with PLOT/DRAW statements. This new "A" shape is in a new file, named appropriately. You build up a suite of character sets, or fonts, and load in the one you need. In PostScript, the fonts are sold on disc or cartridge and you feed these to the printer (or, in some cases, to the computer) so you have onboard the stable of fonts your text calls for.

Yes, yes, yes, but how does all this PostScript shtick work? One bright and sunny day ...

A BOWL OF ASCII SALAD by John Pazmino

... some grunts tapped into the cable connecting the computer to the LaserWriter. What before their wondering eyes spilled out? Out gushed letters! Numbers! A few symbols! And nothing else. They were all pure and simple ASCII characters.

What the hell was this junk!? It was a veritable program. A stream of code. All in a fullblown mature sophisticated language. This was the PostScript language!

The publisher program running on the computer took all the keystrokes and mouseclicks and turned them into a PostScript program in unmitigated ASCII text. It then piped this code to the printer, where it was digested into finished pages. The publisher was an early -- and most successful -- example of CASE, Computer Assisted Software Engineering, a program that writes programs to satisfy your desires. In this situation the desire is a page of true typographic quality and the output program is the PostScrupt code.

Now, if the printer sees only this stream of ASCII chars piping into it, can you not set the publisher aside and write the code yourself? Perhaps with a wordproc or even by typing directly to the printer on the keyboard?

Yes!, you can!!

When the intermedium of the PostScript ASCII textfile was discovered, in 1987, the monopoly of Aldus PageMaker was broken and the LaserWriter lost its aura. In 1989 Adobe threw open the entire specification and many of the internal technica of PostScript for public enjoyment. Today there are scores of applications that can output their design and text as a PostScript ASCII textfile.

You don't need any of these; you can type in your own textfile. It'll look the same to the printer. Just make sure the file is truly pure ASCII, with no funny characters mixed in. On Sinclair, Tasword, for one major example, produces pure ASCII textfiles. This is an excellent vehicle for writing PostScript code. The text within LPRINT statements in BASIC are pure ASCII (provided you do not deliberately insert any hidden characters like BRIGHT or INK).

In examining sample PostScript code you notice a strong resemblance to Forth. Should you be versed in, say, Aberforth or White Lightning, you'll catch on to PostScript very easily. (A hushhush feature of White Lightning is that it can be tweeked into a subset of PostScript!)

PostScript is essentially a stackbased scheme. Parms and args are pushed onto a stack and then commands work on them. The result of the command, if any, is pushed on to the stack in the place of the original parms. PostScript, like Forth, has a fistful of commands to manipulate the stack and query it. You need these for you must at every instant be aware of the stack's contents.

Along with the stack system is the postfix notation. The parms come first, left to right, followed by the command that works on them. PostScript says '130 20 MOVETO' to set the 'pen' to the coord point (130,20); compare with 'PLOT 130,20'.

There are no discrete statements or records in PostScript. The code can run on and on in one entrainment, with linebreaks for your own convenience. However, spaces are critical. Where a space is required between words of code it must be at least one blank or a newline. You may insert many blanks, as to indent or group logical

segments, but PostScript counts all these blanks as one 'whitespace'.

You can make up allnew commands and refer to them in your program, somewhat like subroutines. '/LIST {20 300 MOVETO (Long Island Timex Sinclair) SHOW SHOWPAGE} DEF' captures the stuff '20 ... SHOWPAGE' into a new command LIST. When you do LIST the name of America's premier Sinclair advocancy is printed.

Case is important in that once a command (or variable) is defined with one spelling it must thereafter be refered by that exact same spelling. Also, the names of fonts must be exactly as they are spelled. For the AdvantgePrestige_Bold font, you must say 'AdvantagePrestige_Bold' just like that.

PostScript is truly a general purpose language, the equal of BASIC or Pascal. It has maths, booleans, conditionals, arrays, files, loops, strings, and so on. The emphasis on graphics and text functions makes PostScript a lingua franca for CADs, publishers, wordprocs, artists, illustrators, and other kindred programs. It is the basis of Carousel, the newly adopted medium for video and audio handling in a manner analogous to the massaging of text and graphics.

In fact the character sets themselves, being merely a set of graphics instructions like in the "A" program, are written in PostScript. And, what's more, you can make up your own new sets by coding them in PostScript.

I can not give here even a cursory tuition in PostScript. I do, in the stead, offer a bibliography. You can chew on this until we set out into PostScript territory, two sessions from now.

The litterature of PostScript, quite thoro and rigorous, is still small enough to fit within a ha'meter of bookshelf. By history most of the titles are cited by the color of their covers. PostScript litterati know what you mean by the 'orange book'. Also by history the pressruns of the early titles were short, due to the circumscribed world of PostScript enforced during the Adobe-Aldus-Apple regime. Hence, the older books may be tough to acquire nowadays.

Adobe Systems Inc, "PostScript Language Tutorial and Cookbook", 1985, blue
Adobe Systems Inc, "PostScript Language Reference Manual",1985, red
Adobe Systems Inc, "PostScript Language Program Design", 1988, green
Adobe Systems Inc, "Adobe Type-1 Font Format", 1990, black & white
Braswell, "Inside PostScript", 1989, cloud
Holzgang, "Understanding PostScript Programming", 1987, slate
Holzgang, "PostScript Programming Reference Manual", 1989, charcoal
Kunkel, "Graphic Design in PostScript", 1990, sand
Reid, "Thinking in PostScript", 1990, white
Roth, "Real World PostScript", 1988, orange
Smith, "Learning PostScript", 1990, peach
Thomas, "A PostScript Cookbook", 1988, puke
Wollenweider, "Encapsulated PostScript", 1990, black

There are one or two new titles yearly, as opposed to dozens for just about any other computer topic. The paucity of new offerings is largely the result of the completeness and tightness of the original language and the dominance that Adobe still exerts over it. To maintain the absolute portability of PostScript programs, offAdobe products observe fielty before the specs & regs handed down by Adobe.

Ah, there's a catch here. Like with duck stew, you need ...

(To Be Continued Next Issue!)

My Sinclair can really speak PostScript

This is the continuation of last months article on Success by
Michael E. Carver

```
247 disk_block
365 PRINT #3;'flp1_Success_68000_exe to flp1_Success_68000_exe'
375 DELETE flp1_Success_68000_bak
385 COPY flp1_success_68000_exe,flp1_Success_68000_bak
395 DELEZTE flp1_success_68000_exe
405 PRINT #E=3:'flp1_Success_68000_exe'
415 SEXEC flp1_success_68000_exe,base2,7884,256
475 REMark ***following for use with ramdisk of 720 sectors or more
480 DEFine PROCedure disk_block
490 DIM b(2,3)
500 FOR n=0 TO 3
510 b(0,n)=PEEK_L(base+3270+4*n)
520 b(1,n)=PEEK_L(base+3286+4*n)
530 b(2,n)=PEEK_L(base+3286+4*n)
540 END FOR n
550 IF dev$(0,3)=="mdv": m=1: n_block=3270: move_block
560 if dev$(4,3)=='ram' or dev$(4,3)=='flp': m=0: n_block=3286:
move_block
570 if dev$(8,3)=='flp':m=0:n_block =3302: move_block
580 if dev$(8,3)=='mdv': m=1: n_block=3302: move_block
590 END DEFine disk_block
700 DEFine PROCedure move_block
710 FOR n=0 to 3
720 POKE_L (base+n_block+n*4),b(m,n)
730 END FOR n
740 END DEFine new_block
```

LISTING 2 - Boot program to transfer SUCCESS from floppy to RAM and start SUCCESS from RAM

```
10 REMark ***rename SUCCESS boot to cpm_boot on your master disk***
20 PRINT "To install CPM on RAMdisk >>>"\"key in <CPM>"
30 DEFine PROCedure cpm
40 COPY flp1_cpm_boot,ram1_cpm_boot
50 COPY flp1_success_68000_exe, ram1_success_68000_exe
60 COPY flp1_bios_cde,ram1_bios_cde
70 COPY flp1_cpmfiles,ram1_cpmfiles
80 LRUN ram1_cpm_boot
90 END DEFine cpm
```

LISTing Policy

Annual Dues....$16.00

One 'sample' copy sent upon reciept of a large SASE.

Copies provided on exchange basis with other bona fide user groups.

LISTing is published monthly by LIST (Long Island Sinclair Timex) Group, a non profit user group.

NOTE:
The normal membership year is February through January at a cost of $16.00 By keeping as many members as possible on that basis, we keep our costs and chances of error down. If you wish to begin your subscription later in the year, please sign up for the rest of this year and all of next.

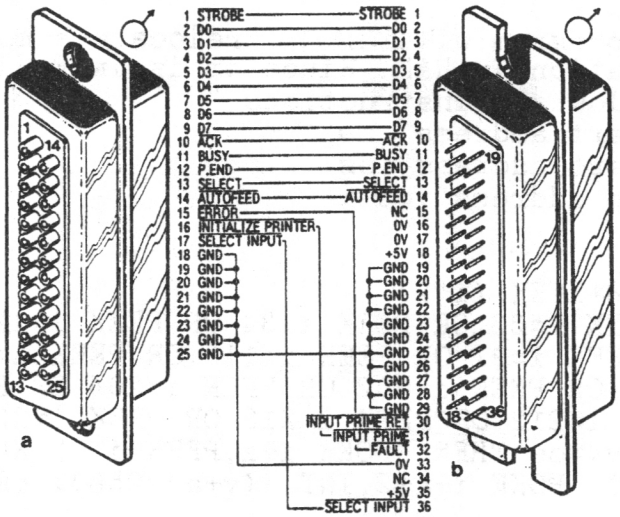
We will accept partial years or different subscription runs, on a limited basis (particularly from members outside the U.S.). BUT, please remember that in addition to possible rate increases, your account must be handled 'by hand' and errors may occur.

POLICY REGARDING CONTRIBUTED MATERIAL:
We are always looking for interesting articles, programs, reviews, etc. to keep our members informed and entertained. Articles submitted for publication are printed on the following basis:
1) you, the writer maintain the full copyright and can resell, lend, or give away your work, as you wish.
2)We are granted the right to publish your material in the original issue in which it appears. Reprints (eg. to supply orders for back issues) will include your work as part of its original issue.
We can't (for now) pay you for your material, but you will recieve a copy of the issue in which it is published, even if you are not a member. You may get more than one issue, and you will definitely earn the respect and appreciation of your grateful peers.

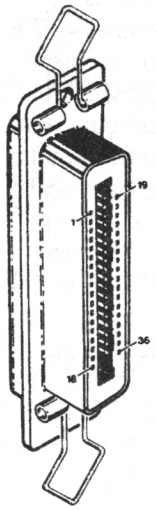
Articles represent the opinion of the author and not necessarily the LIST Group.
LIST dislcaims any responsability for any damage you may do to your computer as a result of reading any articles in LISTing.

Centronics

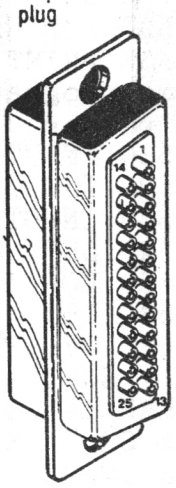
Centronics printer cable connections



36-pin Centronics socket

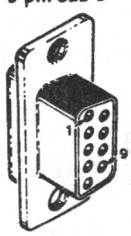


25-pin sub-D plug

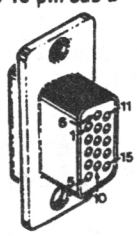


Video

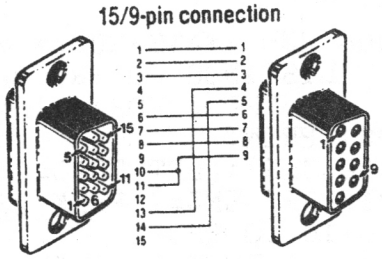
① 9-pin sub-D



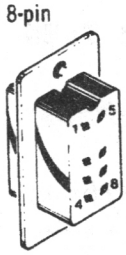
② 15-pin sub-D



15/9-pin connection



③ 8-pin



	MDA ① (Hercules)	CGA ① dig. RGBI	EGA ③ dig. RGBI	NEC ① dig. RGBI	VGA ② an. RGB
1	ground	ground	ground	intensity	R - out
2	ground	ground	R'	R	G - out
3	n.c.	R	R	G	B - out
4	n.c.	G	G	B	mon. ID bit 2
5	n.c.	B	B	ground	ground
6	intensity	intensity	G'	ground	R - return
7	video	n.c.	B'	h - sync (+)	G - return
8	h - sync (+)	h - sync (+)	h - sync (+)	v - sync (-)	B - return
9	v - sync (-)	v - sync (-)	v - sync (-)		(no pin)
10					sync - return
11					mon. ID bit 0
12					mon. ID bit 1
13					h - sync
14					v - sync
15					reserved

This handy pinout was sent to me by one of our members. It shows Parallel printer connectors and IBM video cable connectors.

```
   1 REM THIS TAPE WAS DONATED              BY TORONTO TIMEX
       SINCLAIR USER'S GROUP.
   2 REM MODIFIED AND COPIED BY             ALGIS E. GEDRIS
       DECEMBER 20, 1986
   3 REM  "2068" COMPUTER
9500 REM RELOCATE
9505 PRINT "Note:- This program will        relocate a machine c
ode program to a new location in RAM. Since it is in BASIC, it i
s rather    slow."''"Follow the instructions."
9510 INPUT "Enter new start address",ns
9520 INPUT "Enter old start address",os
9530 INPUT "Enter old end address",oe
9540 LET a=ns-os
9550 FOR i=os TO oe
9560 LET v=PEEK (i+1)+256*PEEK (i+2)
9570 IF (PEEK i=17 OR PEEK i=33 OR PEEK i=34 OR PEEK i=42 OR PEE
K i=50 OR PEEK i=83 OR PEEK i=91 OR PEEK i=115 OR PEEK i=194 OR
PEEK i=195 OR PEEK i=196 OR PEEK i=202 OR PEEK i=204 OR PEEK i=2
05 OR PEEK i=210 OR PEEK i=212 OR PEEK i=218 OR PEEK i=220 OR PE
EK i=242) AND v>=os AND v<=oe THEN  POKE i+a,PEEK i: POKE i+1+a,
(v+a)-256*INT ((v+a)/256): POKE i+a+2,INT ((v+a)/256): LET i=i+2
: GO TO 9590
9580 POKE i+a,PEEK i
9590 NEXT i:: CLEAR (ns-1)
9595 STOP
9998 RANDOMIZE USR 100: SAVE "RELOC.BL" LINE 9500
```

```
9000 REM ***TS-2068***                  UTILITY TO MAKE READING
   LISTINGS OF PROGRAMS EASIER.NOTE:  RETURN OU PROGRAM MUST
      END BEFORE LINE 9000
9001 PRINT '
9010 CLS : LET FLAG=2: LET A=26709
9012 LET END=((PEEK 23627+256*PEEK 23628)-1): GO SUB 9018: FOR A
=26710 TO END-1
9013 IF PEEK A=14 THEN  LET A=A+4: NEXT A
9014 IF PEEK A>31 OR PEEK A=13 THEN  PRINT CHR$ PEEK A;: IF PEEK
 A=13 THEN  PRINT
9015 LET FLAG=FLAG-1: IF PEEK A=13 THEN  GO SUB 9018: LET FLAG=3
: LET A=A+3: PRINT " ";: NEXT A
9016 NEXT A: STOP
9017 REM LINE # SUB
9018 IF A>END-1 THEN  STOP
9019 PRINT PEEK (A+2)+(PEEK (A+1)*256);: RETURN
9998 RANDOMIZE USR 100: SAVE "LLIST.BL" LINE 9000
```